用算法的思想解决一个高考数列题

200231 上海市民办华育中学(华东师范大学数学系教育硕士) 王 江

在上海高中二期课改中算法初步也编进了中学数学教材, 成为高中生必修课程的一部分. 中学数学中的很多问题, 特别是数列中的问题, 常常可以用算法的思想去解决. 而计算机强大的计算功能, 使繁多的重复计算或复杂的运算变得非常简单.

在此举的一例, 是笔者在阅读2007年上海市高考理工类数学卷时, 怀着"可不可以用算法思想解决"的想法, 发现其第20题的解法就可以用程序框图表示. 题目和框图如下.

若有穷数列$a_1, a_2, \cdots, a_n$(n是正整数), 满足$a_1 = a_n$, $a_2 = a_{n-1}$, $\cdots$, $a_n = a_1$, 即$a_i = a_{n-i+1}$(i是正整数, 且$1 \leqslant i \leqslant n$), 就称该数列为"对称数列".

(1) 已知数列$\{b_n\}$是项数为7的对称数列, 且b_1, b_2, b_3, b_4成等差数列, $b_1 = 2$, $b_4 = 11$, 试写出$\{b_n\}$的每一项;

(2) 已知$\{c_n\}$是项数为$2k-1$ ($k \geqslant 1$)的对称数列, 且c_k, c_{k+1}, $\cdots$, c_{2k-1}构成首项为50, 公差为-4的等差数列, 数列$\{c_n\}$的前$2k-1$项和为S_{2k-1}, 则当k为何值时, S_{2k-1}取到最大值? 最大值为多少?

(3) 对于给定的正整数$m > 1$, 试写出所有项数不超过$2m$的对称数列, 使得$1, 2, 2^2, \cdots, 2^{m-1}$成为数列中的连续项; 当$m > 1500$时, 试求其中一个数列的前2008项和$S_{2008}$.

解: (1) $\because b_4 - b_1 = 3d = 9$,

$\therefore d = 3$,

$\therefore b_1 = 2, b_2 = 5, b_3 = 8, b_4 = 11, b_5 = 8, b_6 = 5, b_7 = 2$.

(2) 令$a_1 = c_k$, $a_i = c_{k+i-1}$,

$\because T_{\max} = T_k$

$$\Longleftrightarrow \begin{cases} a_k \geqslant 0, \\ a_{k+1} \leqslant 0, \end{cases}$$

其中, $T_k = a_1 + a_2 + \cdots + a_k$, 且$S_{2k-1} = 2T_k - c_k$,

故当$\begin{cases} a_k \geqslant 0, \\ a_{k+1} \leqslant 0 \end{cases}$时, T_k取到最大,

此时, S_{2k-1}也取到最大.

按图1框图, 经附录[1]的程序运算后, 得到$S_{\max} = S_{25} = 626$, 此时, $k = 13$.

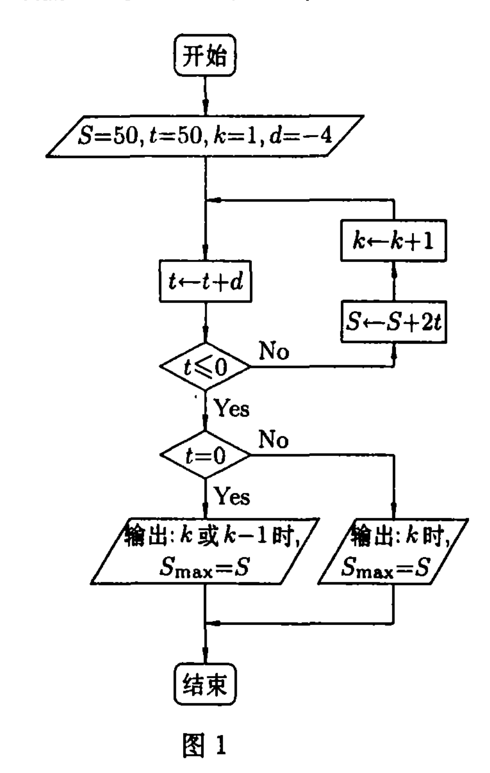


图 1

(3) 满足题意的数列共有四列:

① $1, 2, 2^2, \cdots, 2^{m-2}, 2^{m-1}, 2^{m-2}, \cdots, 2^2, 2, 1$;

② $2^{m-1}, 2^{m-2}, \cdots, 2, 1\ 2, \cdots, 2^{m-2}, 2^{m-1}$;

③ $1, 2, \cdots, 2^{m-2}, 2^{m-1}, 2^{m-1}, 2^{m-2}, \cdots, 2, 1$;

④ $2^{m-1}, 2^{m-2}, \cdots, 2, 1, 1, 2, \cdots, 2^{m-2}, 2^{m-1}$.

对①, 求前2008项和S_{2008}的求法框图如图2.

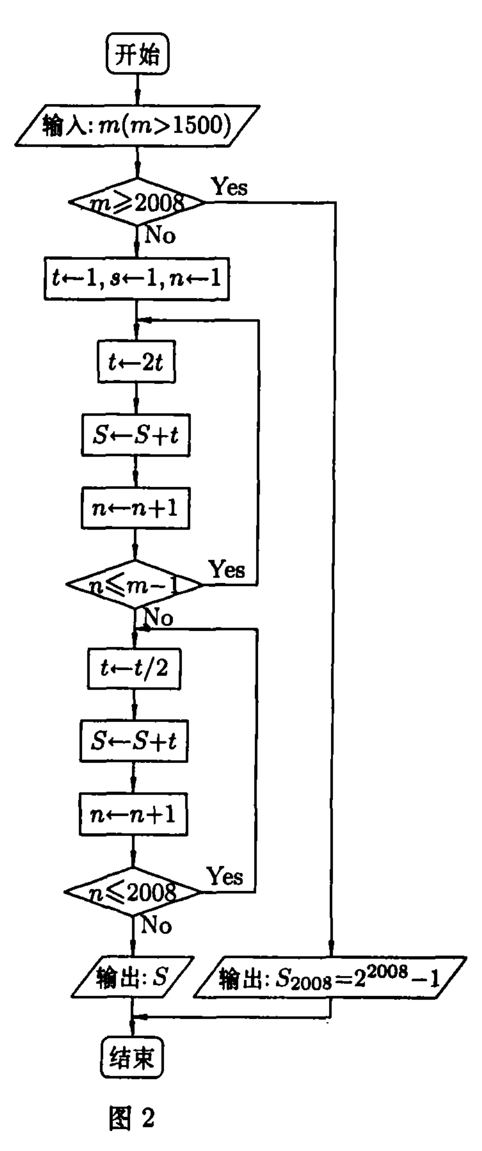


图 2

用按此框图进行编程, 如附录[2], 对于输入大于1500的正整数, 都能由计算机很快计算出 S_{2008} 的值, 这就大大体现出计算机的优越性. 而其中算法实现的过程, 既可以培养学生分步解决问题的思维, 更可以培养学生缜密的逻辑思维.

在平时的学习中, 勤于用算法的思想解决问题, 并用程序框图来表达, 既可以培养学生的逻辑思维能力, 又可以培养学生的表达能力, 而且, 通过这样的思考, 学生可以真正地领会算法思想, 并且养成学以致用的好习惯.

参考文献

[1] 袁震东、赵小平. 数学高二年级第一学期. 上海教育出版社. 2007.

[2] 2007年上海市数学高考命题组. 2007年上海市高考数学理科试题. 上海. 2007.

附录

[1] 第(2)小题的scilab程序:

```
s=50; t=50; d=−4; k=1;
while t>= 4
t=t+d;
s=s+2*t;
k=k+1;
end
if t== 0
disp (s, "时, Smax=", k−1, "或", k, "k=")
else
disp(s, "时, Smax=", k, "k=")
end
```

[2] 第(3)小题的scilab程序:

```
m=input ("输入大于1500的正整数");
if m>=2008
disp("S=2^2008 − 1")
else
t=1; s=1; n=1;
while n<=m
t=2*t;
s=s+t;
n=n+1;
end
while n<=2008
t=t/2;
s=s+t;
n=n+1;
end
disp(s, "S=")
end
```